HỆ THỐNG TẬP TIN

I Mục đích

Sau khi học xong chương này, người học nắm được những kiến thức sau:

- Hiểu các khía cạnh khác nhau của tập tin và cấu trúc thư mục
- Hiểu các cơ chế quản lý, kiểm soát, bảo vệ tập tin khi có nhiều người cùng truy xuất
- Hiểu cách chia sẻ tập tin giữa nhiều quá trình, người dùng và máy tính

II Giới thiệu

Đối với hầu hết người dùng, hệ thống tập tin là diện mạo dễ nhìn thấy nhất của hệ điều hành. Nó cung cấp cơ chế cho việc lưu trữ trực tuyến và truy xuất dữ liệu, chương trình của hệ điều hành và tất cả người dùng của hệ thống máy tính. Hệ thống tập tin chứa hai phần riêng biệt: tập hợp ***các tập tin*** (files), mỗi tập tin lưu trữ dữ liệu có liên quan và ***cấu trúc thư mục*** (directory structure) mà nó tổ chức và cung cấp thông tin về tất cả tập tin trong hệ thống. Một số hệ thống tập tin còn có thêm phần thứ ba , ***các phân khu*** (partitions) mà nó được dùng để tách rời tập hợp các thư mục lớn luận lý và vật lý.

Trong chương này chúng ta xét các khía cạnh khác nhau của tập tin và cấu trúc thư mục. Chúng ta cũng thảo luận các cách để quản lý ***việc bảo vệ tập tin*** (file protection), cần thiết khi nhiều người dùng truy xuất các tập tin và chúng ta muốn kiểm soát ai và cách gì truy xuất tập tin. Cuối cùng, chúng ta thảo luận việc chia sẻ giữa nhiều quá trình, người dùng, và máy tính.

III Khái niệm tập tin

Các máy tính lưu trữ thông tin trên nhiều phương tiện lưu trữ khác nhau, như đĩa từ, băng từ, đĩa quang. Để hệ thống máy tính tiện dụng, hệ điều hành cung cấp một tầm nhìn luận lý không đổi của việc lưu trữ thông tin. Hệ điều hành trừu tượng từ các thuộc tính vật lý của các thiết bị lưu trữ của nó đến định nghĩa một đơn vị lưu trữ luận lý là **tập tin** (file). Tập tin được ánh xạ bởi hệ điều hành trên các thiết bị vật lý. Các thiết bị lưu trữ được dùng thường ổn định vì thế nội dung không bị mất khi mất điện hay khởi động lại hệ thống.

Một tập tin là một tập thông tin có liên quan được ghi trên thiết bị lưu trữ phụ. Từ quan điểm người dùng, một tập tin là phần nhỏ nhất của thiết bị lưu trữ phụ luận lý; nghĩa là dữ liệu không thể được viết tới thiết bị lưu trữ phụ trừ khi chúng ở trong một tập tin. Các tập tin dữ liệu có thể là số, chữ, ký tự số hay nhị phân. Các tập tin có thể có dạng bất kỳ như tập tin văn bản, hay có thể được định dạng không đổi. Thông thường, một tập tin là một chuỗi các bits, bytes, dòng hay mẫu tin,..được định nghĩa bởi người tạo ra nó. Do đó, khái niệm tập tin là cực kỳ tổng quát.

Thông tin trong một tập tin được định nghĩa bởi người tạo. Nhiều loại thông tin khác nhau có thể được lưu trữ trong một tập tin-chương trình nguồn, chương trình đối tượng, chương trình thực thi, dữ liệu số, văn bản, mẫu tin, hình ảnh đồ hoạ, âm thanh,..Một tập tin có một cấu trúc được định nghĩa cụ thể dựa theo loại của nó. Một tập tin văn bản là một chuỗi các ký tự được tổ chức thành những dòng. Một tập tin

nguồn là một chuỗi các thủ tục và hàm, được tổ chức khi khai báo được theo sau bởi các câu lệnh có thể thực thi. Một tập tin đối tượng là một chuỗi các bytes được tổ chức thành các khối có thể hiểu được bởi bộ liên kết của hệ thống. Một tập tin có thể thực thi là một chuỗi các phần mã mà bộ nạp có thể mang vào bộ nhớ và thực thi.

III.1 Thuộc tính tập tin

Để tiện cho người dùng, một tập tin được đặt tên và được tham khảo bởi tên của nó. Một tên thường là một chuỗi các ký tự, thí dụ: example.c. Một số hệ thống có sự phân biệt giữa ký tự hoa và thường trong tên, ngược lại các hệ thống khác xem hai trường hợp đó là tương đương. Khi một tập tin được đặt tên, nó trở nên độc lập với quá trình, người dùng, và thậm chí với hệ thống tạo ra nó. Thí dụ, một người dùng có thể tạo tập tin example.c, ngược lại người dùng khác có thể sửa tập tin đó bằng cách xác định tên của nó. Người sở hữu tập tin có thể ghi tập tin tới đĩa mềm, gởi nó vào email hay chép nó qua mạng và có thể vẫn được gọi example.c trên hệ thống đích.

Một tập tin có một số thuộc tính khác mà chúng rất khác nhau từ một hệ điều hành này tới một hệ điều hành khác, nhưng điển hình chúng gồm:

- **Tên** (name): tên tập tin chỉ là thông tin được lưu ở dạng mà người dùng có thể đọc
- **Định danh** (identifier): là thẻ duy nhất, thường là số, xác định tập tin trong hệ thống tập tin; nó là tên mà người dùng không thể đọc
- **Kiểu** (type): thông tin này được yêu cầu cho hệ thống hỗ trợ các kiểu khác nhau
- **Vị trí** (location): thông tin này là một con trỏ chỉ tới một thiết bị và tới vị trí tập tin trên thiết bị đó.
- **Kích thước** (size): kích thước hiện hành của tập tin (tính bằng byte, word hay khối) và kích thước cho phép tối đa chứa trong thuộc tính này.
- **Giờ** (time), **ngày** (date) và **định danh người dùng** (user identification): thông tin này có thể được lưu cho việc tạo, sửa đổi gần nhất, dùng gần nhất. Dữ liệu này có ích cho việc bảo vệ, bảo mật, và kiểm soát việc dùng.

Thông tin về tất cả tập tin được giữ trong cấu trúc **thư mục** (directory) nằm trong thiết bị lưu trữ phụ. Điển hình, mục từ thư mục chứa tên tập tin và định danh duy nhất của nó. Định danh lần lượt xác định thuộc tính tập tin khác. Trong hệ thống có nhiều tập tin, kích thước của chính thư mục có thể là Mbyte. Bởi vì thư mục giống tập tin, phải bền, chúng phải được lưu trữ trên thiết bị và mang vào bộ nhớ khi cần.

III.2 Thao tác tập tin

Tập tin là ***kiểu dữ liệu trừu tượng***. Để định nghĩa một tập tin hợp lý, chúng ta cần xem xét các thao tác có thể được thực hiện trên các tập tin. Hệ điều hành cung cấp lời gọi hệ thống để thực hiện các thao tác này

- **Tạo tập tin**: hai bước cần thiết để tạo một tập tin. Thứ nhất, không gian trong hệ thống tập tin phải được tìm cho tập tin. Thứ hai, một mục từ cho tập tin mới phải được tạo trong thư mục. Mục từ thư mục ghi tên tập tin và vị trí trong hệ thống tập tin, và các thông tin khác.
- **Mở**: trước khi mở tập tin, quá trình phải mở nó. Mục tiêu của việc mở là cho phép hệ thống thiết lập một số thuộc tính và địa chỉ đĩa trong bộ nhớ để tăng tốc độ truy xuất.

- **Đóng**: khi chấm dứt truy xuất, thuộc tính và địa chỉ trên đĩa không còn dùng nữa, tập tin được đóng lại để giải phóng vùng nhớ.
- **Ghi**: để ghi một tập tin, chúng ta thực hiện lời gọi hệ thống xác định tên tập tin và thông tin được ghi tới tập tin. Với tên tập tin, hệ thống tìm thư mục để xác định vị trí của tập tin. Hệ thống phải giữ một con trỏ viết tới vị trí trong tập tin nơi mà thao tác viết tiếp theo sẽ xảy ra. Con trỏ viết phải được cập nhật bất cứ khi nào thao tác viết xảy ra.
- **Chèn cuối**: giống thao tác ghi nhưng dữ liệu luôn được ghi vào cuối tập tin
- **Đọc**: để đọc từ một tập tin, chúng ta dùng lời gọi hệ thống xác định tên tập tin và nơi (trong bộ nhớ) mà khối tiếp theo của tập tin được đặt. Thư mục được tìm mục từ tương ứng và hệ thống cần giữ con trỏ đọc tới vị trí trong tập tin nơi thao tác đọc tiếp theo xảy ra.
- **Xoá**: để xoá một tập tin, chúng ta tìm kiếm thư mục với tên tập tin được cho. Tìm mục từ tương ứng, giải phóng không gian tập tin để không gian này có thể dùng lại bởi tập tin khác và xoá mục từ thư mục.
- **Tìm**: thư mục được tìm mục từ tương ứng và vị trí con trỏ hiện hành được đặt tới giá trị được cho
- **Lấy thuộc tính**: lấy thuộc tính tập tin cho quá trình
- **Đổi tên**: thay đổi tên tập tin đã tồn tại

III.3 Các kiểu tập tin

Khi thiết kế một hệ thống tập tin, chúng ta luôn luôn xem xét hệ điều hành nên tổ chức và hỗ trợ các kiểu tập tin nào. Nếu hệ điều hành nhận biết kiểu của một tập tin, nó có thể thao tác trên tập tin đó trong các cách phù hợp.

Một kỹ thuật chung cho việc cài đặt các kiểu tập tin là chứa kiểu đó như một phần của tên tập tin. Tên tập tin được chia làm hai phần-tên và phần mở rộng, thường được ngăn cách bởi dấu chấm. Trong trường hợp này, người dùng và hệ điều hành có thể biết kiểu tập tin là gì từ tên.

Các hệ điều hành thường hỗ trợ các kiểu tập tin sau:

- **Tập tin thường**: là tập tin văn bản hay tập tin nhị phân chứa thông tin của người sử dụng
- **Thư mục**: là những tập tin hệ thống dùng để lưu giữ cấu trúc của hệ thống tập tin
- **Tập tin có ký tự đặc biệt**: liên quan đến nhập/xuất thông qua các thiết bị nhập/xuất tuần tự như màn hình, máy in,..
- **Tập tin khối**: dùng để truy xuất trên thiết bị đĩa

III.4 Cấu trúc tập tin

Các kiểu tập tin cũng có thể được dùng để hiển thị cấu trúc bên trong của một tập tin. Ngoài ra, các tập tin cụ thể phải phù hợp cấu trúc được yêu cầu để hệ điều hành có thể hiểu. Một số hệ điều hành mở rộng ý tưởng này thành tập hợp các cấu trúc tập tin được hỗ trợ hệ thống, với những tập hợp thao tác đặc biệt cho việc thao tác các tập tin với những cấu trúc đó.

Các hệ điều hành thường hỗ trợ ba cấu trúc tập tin thông dụng là:

- **Không có cấu trúc**: tập tin là một dãy tuần tự các byte
- **Có cấu trúc**: tập tin là một dãy các mẫu tin có kích thước cố định

- **Cấu trúc cây**: tập tin gồm một cây của những mẫu tin không cần thiết có cùng chiều dài, mỗi mẫu tin có một trường khoá giúp việc tìm kiếm nhanh hơn

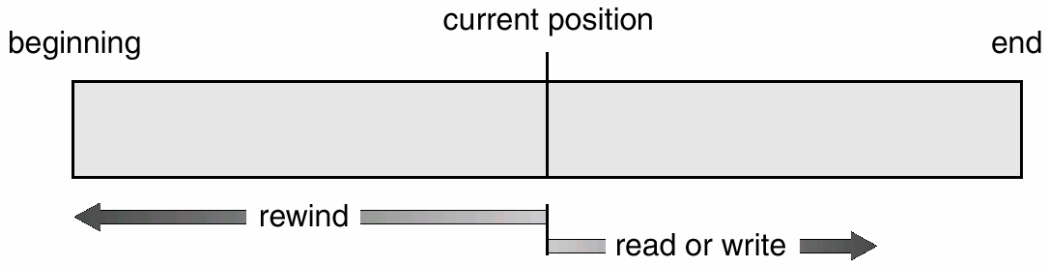
IV Các phương pháp truy xuất

Các tập tin lưu trữ thông tin. Khi nó được dùng, thông tin này phải được truy xuất và đọc vào bộ nhớ máy tính. Thông tin trong tập tin có thể được truy xuất trong nhiều cách.

IV.1 Truy xuất tuần tự

Một phương pháp đơn giản nhất là truy xuất tuần tự. Thông tin trong tập tin được xử lý có thứ tự, một mẫu tin này sau mẫu tin kia. Chế độ truy xuất này là thông dụng nhất. Thí dụ, bộ soạn thảo và biên dịch thường truy xuất các tập tin trong cách thức này.

Nhóm các thao tác trên một tập tin là đọc và viết. Một thao tác đọc đọc phần tiếp theo của tập tin và tự động chuyển con trỏ tập tin để ghi vết vị trí nhập/xuất. Tương tự, một thao tác viết chèn vào cuối tập tin và chuyển tới vị trí cuối của tài liệu vừa được viết (cuối tập tin mới). Trên một vài hệ thống, một tập tin như thế có thể được đặt lại tới vị trí bắt đầu và một chương trình có thể nhảy tới hay lùi n mẫu tin. Truy xuất tuần tự được mô tả như hình IX-1.



Hình 0-1 Truy xuất tập tin tuần tự

IV.2 Truy xuất trực tiếp

Một phương pháp khác là truy xuất trực tiếp (hay truy xuất tương đối). Một tập tin được hình thành từ các mẫu tin luận lý có chiều dài không đổi. Các mẫu tin này cho phép người lập trình đọc và viết các mẫu tin nhanh chóng không theo thứ tự. Phương pháp truy xuất trực tiếp dựa trên mô hình đĩa của tập tin, vì đĩa cho phép truy xuất ngẫu nhiên tới bất cứ khối tập tin. Để truy xuất trực tiếp, tập tin được hiển thị như một chuỗi các khối hay mẫu tin được đánh số. Tập tin truy xuất trực tiếp cho phép các khối bất kỳ được đọc hay viết. Do đó, chúng ta có thể đọc khối 14, sau đó đọc khối 53 và sau đó viết khối 7. Không có bất kỳ sự hạn chế nào trên thứ tự đọc hay viết cho một tập tin truy xuất trực tiếp

Các tập tin truy xuất trực tiếp được dùng nhiều cho truy xuất tức thời tới một lượng lớn thông tin. Cơ sở dữ liệu thường là loại này. Khi một truy vấn tập trung một chủ đề cụ thể, chúng ta tính khối nào chứa câu trả lời và sau đó đọc khối đó trực tiếp để cung cấp thông tin mong muốn.

Không phải tất cả hệ điều hành đều hỗ trợ cả hai truy xuất tuần tự và trực tiếp cho tập tin. Một số hệ thống cho phép chỉ truy xuất tập tin tuần tự; một số khác cho

phép chỉ truy xuất trực tiếp. Một số hệ điều hành yêu cầu một tập tin được định nghĩa như tuần tự hay trực tiếp khi nó được tạo ra; như tập tin có thể được truy xuất chỉ trong một cách không đổi với khai báo của nó. Tuy nhiên, chúng ta dễ dàng mô phỏng truy xuất tuần tự trên tập tin truy xuất trực tiếp. Nếu chúng ta giữ một biến ***cp*** để xác định vị trí hiện tại thì chúng ta có thể mô phỏng các thao tác tập tin tuần tự như được hiển thị trong hình IX-2. Mặc dù, không đủ và không gọn để mô phỏng một tập tin truy xuất trực tiếp trên một tập tin truy xuất tuần tự.

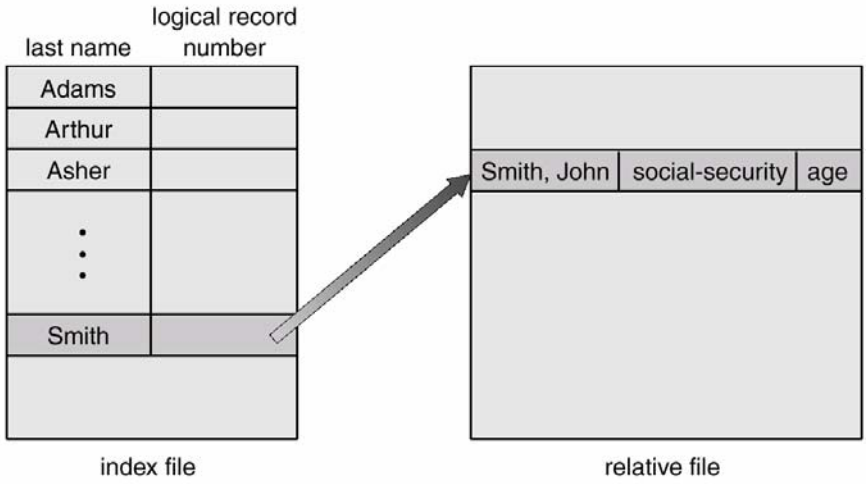
sequential access	implementation for direct access
reset	*cp* = 0;
read next	*read cp*; *cp* = *cp*+1;
write next	*write cp*; *cp* = *cp*+1;

Hình 0-2 Mô phỏng truy xuất tuần tự trên truy xuất trực tiếp

IV.3 Các phương pháp truy xuất khác

Các phương pháp truy xuất khác có thể được xây dựng trên cơ sở của phương pháp truy xuất trực tiếp. Các phương pháp khác thường liên quan đến việc xây dựng chỉ mục cho tập tin. Chỉ mục chứa các con trỏ chỉ tới các khối khác. Để tìm một mẫu tin trong tập tin, trước hết chúng ta tìm chỉ mục và sau đó dùng con trỏ để truy xuất tập tin trực tiếp và tìm mẫu tin mong muốn.

Với những tập tin lớn, chỉ mục tập tin có thể trở nên quá lớn để giữ trong bộ nhớ. Một giải pháp là tạo chỉ mục cho tập tin chỉ mục. Tập tin chỉ mục chính chứa các con trỏ chỉ tới các tập tin chỉ mục thứ cấp mà nó chỉ tới các thành phần dữ liệu thật sự.



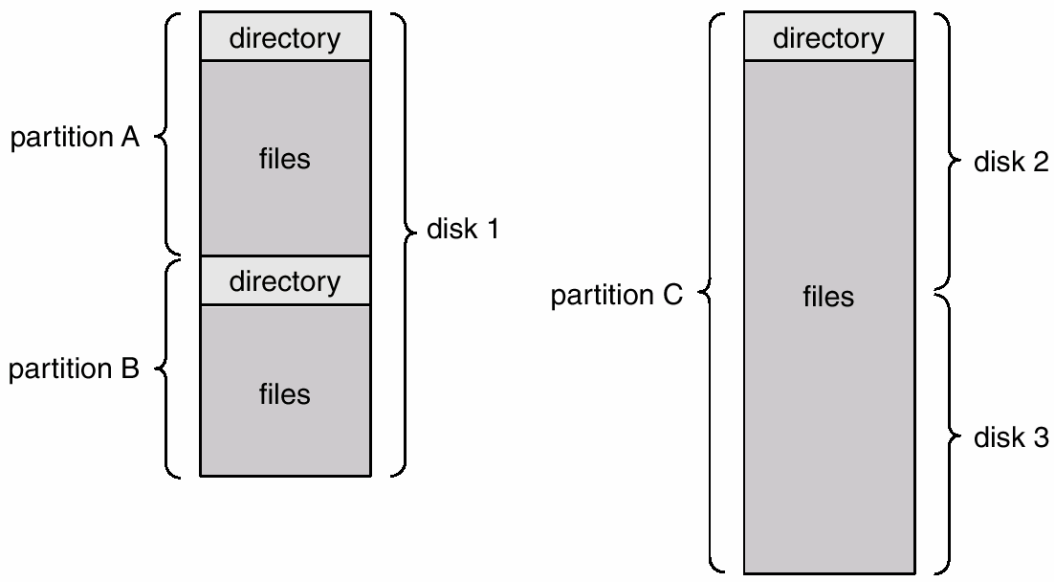
Hình 0-3 Thí dụ về chỉ mục và các tập tin liên quan

V Cấu trúc thư mục

Các hệ thống tập tin của máy tính có thể rất lớn về số lượng. Một số hệ thống lưu trữ hàng triệu tập tin trên các terabytes đĩa. Để quản lý tất cả dữ liệu này, chúng ta cần tổ chức lại chúng. Việc tổ chức này thường được thực hiện hai phần.

Thứ nhất, đĩa được chia thành một hay nhiều phân khu (partition) hay phân vùng (volumes). Điển hình, mỗi đĩa trên hệ thống chứa ít nhất một phân khu. Phân khu này là cấu trúc cấp thấp mà các tập tin và thư mục định vị. Thỉnh thoảng các phân khu được dùng để cung cấp nhiều vùng riêng rẻ trong một đĩa, mỗi phân khu được xem như một thiết bị lưu trữ riêng, trái lại các hệ thống khác cho phép các phân khu có dung lượng lớn hơn một đĩa để nhóm các đĩa vào một cấu trúc luận lý và cấu trúc tập tin, và có thể bỏ qua hoàn toàn những vấn đề cấp phát không gian vật lý cho các tập tin. Cho lý do này, các phân khu có thể được xem như các đĩa ảo. Các phân khu cũng có thể lưu trữ nhiều hệ điều hành, cho phép hệ thống khởi động và chạy nhiều hơn một hệ điều hành.

Thứ hai, mỗi phân khu chứa thông tin về các tập tin trong nó. Thông tin này giữ trong những mục từ trong một thư mục thiết bị hay bảng mục lục phân vùng (volume table of contents). Thư mục thiết bị (được gọi đơn giản là thư mục) ghi thông tin-như tên, vị trí, kích thước và kiểu-đối với tất cả tập tin trên phân khu (như hình IX-4).



Hình 0-4 tổ chức hệ thống tập tin điển hình

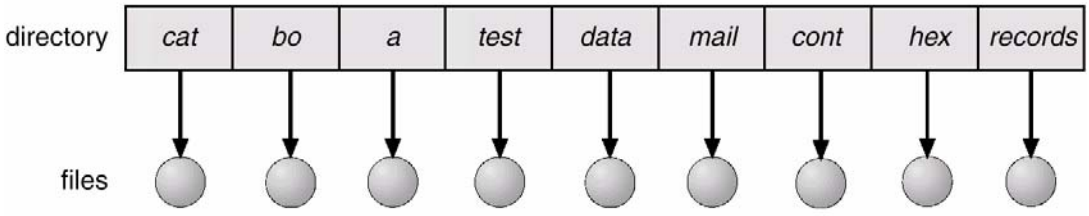
Thư mục có thể được hiển thị như một bảng danh biểu dịch tên tập tin thành các mục từ thư mục. Các thư mục có thể được tổ chức trong nhiều cách. Chúng ta muốn có thể chèn mục từ, xoá mục từ, tìm kiếm một mục từ và liệt kê tất cả mục từ trong thư mục. Trong phần này, chúng ta xem xét nhiều cơ chế định nghĩa cấu trúc luận lý của hệ thống thư mục. Khi xem xét một cấu trúc thư mục cụ thể, chúng ta cần nhớ các thao tác được thực hiện trên một thư mục.

- **Tìm kiếm tập tin**: chúng ta cần tìm trên cấu trúc thư mục để xác định mục từ cho một tập tin cụ thể.
- **Tạo tập tin**: một tập tin mới cần được tạo và được thêm tới thư mục.
- **Xoá tập tin**: khi một tập tin không còn cần, chúng ta muốn xoá nó ra khỏi thư mục.
- **Liệt kê thư mục**: chúng ta có thể liệt kê các tập tin trong thư mục và nội dung của mục từ thư mục cho mỗi tập tin trong danh sách.
- **Đổi tên tập tin**: vì tên tập tin biểu diễn nội dung của nó đối với người dùng, tên có thể thay đổi khi nội dung hay việc sử dụng tập tin thay đổi. Đổi tên tập tin có thể cho phép vị trí của nó trong cấu trúc thư mục được thay đổi.
- **Duyệt hệ thống tập tin**: chúng ta muốn truy xuất mỗi thư mục và mỗi tập tin trong cấu trúc thư mục.

Chúng ta sẽ mô tả các cơ chế thông dụng nhất để định nghĩa cấu trúc luận lý của một thư mục.

V.1 Cấu trúc thư mục dạng đơn cấp

Cấu trúc thư mục đơn giản nhất là thư mục đơn cấp. Tất cả tập tin được chứa trong cùng thư mục như được hiển thị trong hình IX-5 dưới đây:



Hình 0-5 Thư mục đơn cấp

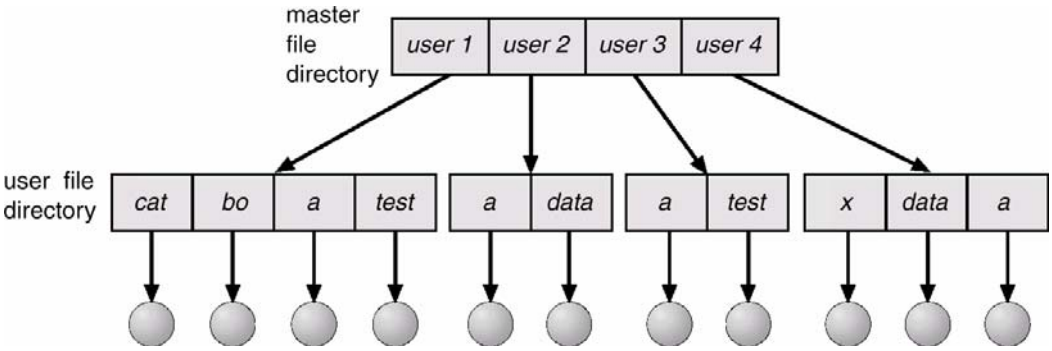
Tuy nhiên, thư mục đơn cấp có nhiều hạn chế khi số lượng tập tin tăng hay khi hệ thống có nhiều hơn một người dùng. Vì tất cả tập tin được chứa trong cùng thư mục, chúng phải có tên khác nhau. Nếu hai người dùng đặt tên tập tin dữ liệu của họ là test thì qui tắc tên duy nhất bị xung đột. Mặc dù các tên tập tin thường được chọn để phản ánh nội dung của tập tin, chúng thường bị giới hạn chiều dài. Hệ điều hành MS-DOS cho phép chỉ 11 ký tự cho tên; UNIX cho phép 255 ký tự.

Người dùng trên thư mục đơn cấp có thể gặp phải khó khăn để nhớ tên của tất cả tập tin, khi số tập tin tăng. Nếu trên một hệ thống máy tính có hàng trăm tập tin thì việc ghi lại vết của quá nhiều tập tin là một tác vụ nặng nề.

V.2 Cấu trúc thư mục dạng hai cấp

Một thư mục đơn cấp dẫn đến sự lẫn lộn giữa tên các tập tin của nhiều người dùng khác nhau. Giải pháp chuẩn là tạo một thư mục riêng cho mỗi người dùng.

Trong cấu trúc thư mục hai cấp, mỗi người dùng có thư mục tập tin riêng cho họ (user file directory-UFD). Mỗi UFD có một cấu trúc tương tự nhưng các danh sách chứa các tập tin của một người dùng. Khi công việc của người dùng bắt đầu hay người dùng đăng nhập, thư mục tập tin chính của hệ thống (master file directory) được tìm kiếm. MFD được lập chỉ mục bởi tên người dùng hay số tài khoản và mỗi mục từ chỉ tới UFD cho người dùng đó (như hình IX-6)



Hình 0-6 thư mục hai cấp

Khi người dùng tham khảo tới một tập tin cụ thể, chỉ UFD của chính người dùng đó được tìm kiếm. Do đó, các người dùng khác nhau có thể có các tập tin với cùng một tên, với điều kiện là tất cả tên tập tin trong mỗi UFD là duy nhất.

Để tạo một tập tin cho một người dùng, hệ điều hành chỉ tìm UFD của người dùng đó để xác định một tập tin khác cùng tên có tồn tại hay không. Để xóa một tập tin, hệ điều hành giữ lại việc tìm kiếm của nó tới UFD cục bộ; do đó, nó không thể xóa nhằm tập tin của người dùng khác có cùng tên.

Các thư mục người dùng phải được tạo và xóa khi cần thiết. Một chương trình hệ thống đặc biệt được chạy với tên người dùng hợp lý và thông tin tài khoản. Chương trình này tạo một UFD mới và thêm một mục từ cho nó tới MFD. Việc thực thi chương trình này có thể bị giới hạn bởi người quản trị hệ thống.

Mặc dù cấu trúc thư mục hai cấp giải quyết vấn đề xung đột tên nhưng nó cũng có những bất lợi. Cấu trúc này cô lập một người dùng từ người dùng khác. Việc cô lập này là lợi điểm khi các người dùng hoàn toàn độc lập nhau nhưng sẽ bất lợi khi các người dùng muốn hợp tác trên một số công việc và để truy xuất các tập tin của người dùng khác. Một số hệ thống đơn giản không cho phép tập tin người dùng cục bộ được truy xuất bởi người dùng khác.

Nếu truy xuất được cho phép, một người dùng phải có khả năng đặt tên một tập tin trong một thư mục của người dùng khác. Để đặt tên một tập tin xác định duy nhất trong thư mục hai cấp, chúng ta phải cho cả hai tên người dùng và tên tập tin. Một thư mục hai cấp có thể được xem như một cây hay ít nhất một cây đảo ngược hay có chiều cao bằng 2. Gốc của cây là UFD. Hậu duệ trực tiếp của nó là MFD. Hậu duệ của UFD là các tập tin. Các tập tin này là lá của cây. Xác định tên người dùng và tên tập tin định nghĩa đường dẫn trong cây từ gốc (MFD) tới một lá (tập tin xác định). Do đó, tên người dùng và tên tập tin định nghĩa tên đường dẫn. Mọi tập tin trong hệ thống có một đường dẫn. Để đặt tên một tập tin duy nhất người dùng phải biết tên đường dẫn của tập tin mong muốn.

Trường hợp đặc biệt xảy ra cho các tập tin hệ thống. Các chương trình này cung cấp một phần hệ thống như: bộ nạp, bộ hợp ngữ, bộ biên dịch, các thủ tục,..thường được định nghĩa như các tập tin. Khi các lệnh tương ứng được gọi tới hệ điều hành, các tập tin này được đọc bởi bộ nạp và được thực thi. Một số bộ thông dịch lệnh hoạt động bằng cách xem lệnh như là tên tập để nạp và thực thi. Với hệ thống thư mục được định nghĩa hiện tại, tên tập tin này được tìm kiếm trong UFD hiện hành. Một giải pháp cho vấn đề này là chép các tập tin hệ thống vào mỗi UFD. Tuy nhiên, chép tất cả tập tin hệ thống sẽ lãng phí lượng lớn không gian.

Giải pháp chuẩn là làm phức tạp thủ tục tìm kiếm một chút. Một thư mục người dùng đặc biệt được định nghĩa để chứa các tập tin hệ thống (thí dụ, user0). Bất cứ khi nào một tên tập tin được cho để được nạp, trước tiên hệ điều hành tự tìm thư mục người dùng cục bộ. Nếu không tìm thấy, hệ điều hành tự tìm trong thư mục

người dùng đặc biệt này. Một chuỗi các thư mục được tìm khi một tập tin được đặt tên là **đường dẫn tìm kiếm**. Ý tưởng này có thể được mở rộng, đường dẫn tìm kiếm chứa danh sách các thư mục không giới hạn để tìm khi tên một lệnh được cho. Phương pháp này được dùng nhiều nhất trong UNIX và MS-DOS.

V.3 Cấu trúc thư mục dạng cây

Thư mục cấu trúc cây là trường hợp tổng quát của thư mục hai cấp. Sự tổng quát này cho phép người dùng tạo thư mục con và tổ chức các tập tin của họ. Thí dụ, hệ thống MS-DOS có cấu trúc cây. Thật vậy, một cây là cấu trúc thư mục phổ biến nhất. Cây có thư mục gốc. Mỗi tập tin trong hệ thống có tên đường dẫn duy nhất. Tên đường dẫn là đường dẫn từ gốc xuống tất cả thư mục con tới tập tin xác định.

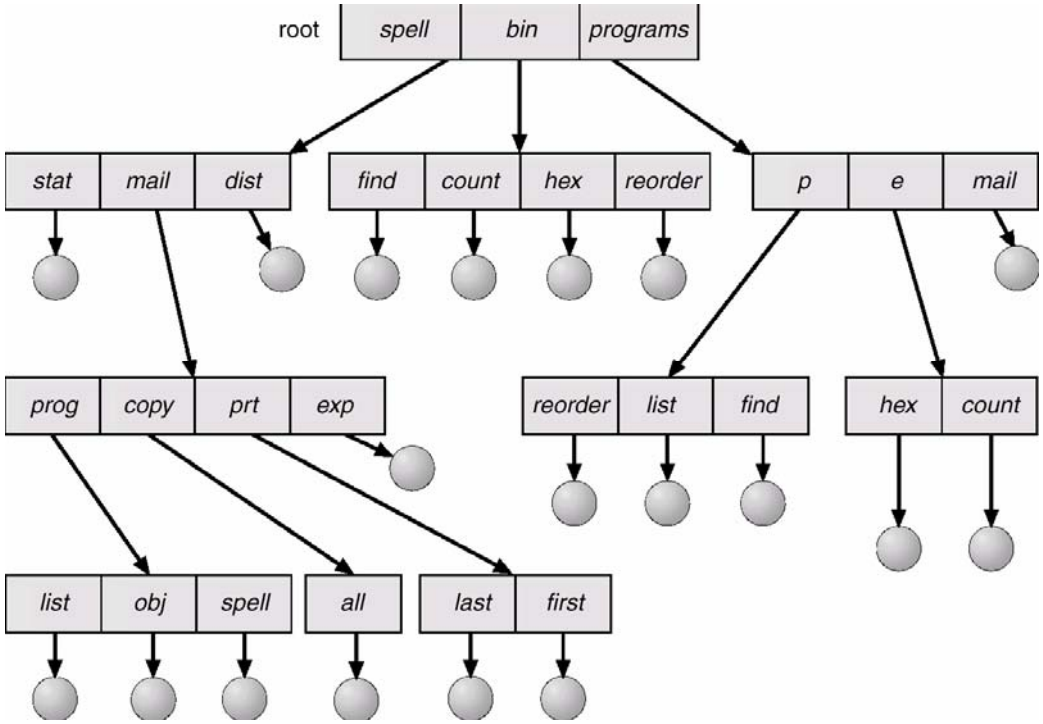
Một thư mục (hay thư mục con) chứa tập hợp các tập tin hay thư mục con. Một thư mục đơn giản là tập tin nhưng nó được đối xử trong một cách đặc biệt. Tất cả thư mục có cùng định dạng bên trong. Một bit trong mỗi mục từ thư mục định nghĩa mục từ như một tập tin (0) hay như một thư mục con (1). Các lời gọi hệ thống đặc biệt được dùng để tạo và xoá thư mục.

Thường thì mỗi người dùng có ***thư mục hiện hành***. Thư mục hiện hành chứa hầu hết các tập tin người dùng hiện đang quan tâm. Khi tham khảo được thực hiện tới tập tin, thư mục hiện hành được tìm. Nếu một tập tin được yêu cầu mà nó không có trong thư mục hiện hành thì người dùng phải xác định tên đường dẫn hay chuyển thư mục hiện hành tới thư mục quản lý tập tin đó. Để thay đổi thư mục, một lời gọi hệ thống được cung cấp kèm theo tên thư mục như là tham số và dùng nó để định nghĩa lại thư mục hiện hành. Do đó, người dùng có thể thay đổi thư mục hiện hành bất cứ khi nào người dùng muốn.

Thư mục hiện hành khởi đầu của người dùng được gán khi công việc người dùng bắt đầu hay người dùng đăng nhập vào hệ thống. Hệ điều hành tìm tập tin tính toán để xác định mục từ cho người dùng này. Trong tập tin tính toán là con trỏ chỉ tới thư mục khởi đầu của người dùng. Con trỏ này được chép tới một biến cục bộ cho người dùng xác định thư mục hiện hành khởi đầu.

Tên đường dẫn có hai kiểu: ***tên đường dẫn tuyệt đối*** và ***tên đường dẫn tương đối***. Một đường dẫn tuyệt đối bắt đầu từ gốc và theo sau là đường dẫn xuống tới tập tin xác định, cho tên các thư mục trên đường dẫn. Tên đường dẫn tương đối định nghĩa một đường dẫn từ thư mục hiện hành. Thí dụ, trong hệ thống tập tin có cấu trúc cây như hình IX-7, nếu thư mục hiện hành là *root/spell/mail* thì tên đường dẫn tương đối *prt/first* tham chiếu tới cùng tập tin nhưng tên đường dẫn tuyệt đối *root/spell/mail/prt/first*.

Quyết định một chính sách trong cấu trúc thư mục cây là cách để quản lý việc xoá một thư mục. Nếu một thư mục rỗng, mục từ của nó trong thư mục chứa bị xoá. Tuy nhiên, giả sử thư mục bị xoá không rỗng, nhưng chứa nhiều tập tin và thư mục con; một trong hai tiếp cận có thể được thực hiện. Một số hệ thống như MS-DOS sẽ không xoá một thư mục nếu nó không rỗng. Do đó, để xoá một thư mục, người dùng trước hết phải xoá tất cả tập tin trong thư mục đó. Nếu bất cứ thư mục con tồn tại, thủ tục này phải được áp dụng đệ qui tới chúng để mà chúng có thể bị xoá. Tiếp cận này dẫn đến lượng công việc lớn.



Hình 0-7 cấu trúc thư mục dạng cây

Một tiếp cận khác được thực hiện bởi lệnh **rm** của UNIX cung cấp tuỳ chọn mà khi một yêu cầu được thực hiện để xoá một thư mục, tất cả tập tin và thư mục con của thư mục đó cũng bị xoá. Tiếp cận này tương đối đơn giản để cài đặt; chọn lựa này là một chính sách. Chính sách sau đó tiện dụng hơn nhưng nguy hiểm hơn vì toàn bộ cấu trúc thư mục có thể bị xoá với một lệnh. Nếu lệnh đó được cấp phát bị lỗi, một số lượng lớn tập tin và thư mục cần được phục hồi từ các băng từ sao lưu.

Với một hệ thống thư mục cấu trúc cây, người dùng có thể truy xuất tới các tập tin của họ và các tập tin của người dùng khác. Thí dụ, người dùng B có thể truy xuất các tập tin của người dùng A bằng cách xác định tên đường dẫn của chúng. Người dùng B có thể xác định tên đường dẫn tương đối hay tuyệt đối. Người dùng B có thể chuyển thư mục hiện hành tới thư mục của người dùng A và truy xuất các tập tin bằng tên của chúng. Một số hệ thống cũng cho phép người dùng định nghĩa đường dẫn tìm kiếm của chính họ. Trong trường hợp này, người dùng B có thể định nghĩa đường dẫn tìm kiếm của mình là (1) thư mục cục bộ của mình, (2) thư mục tập tin hệ thống và (3) thư mục của người dùng A, theo thứ tự đó. Tập tin có thể được tham khảo đơn giản bằng tên với điều kiện tên tập tin của người dùng A không xung đột với tên của một tập tin cục bộ hay tập tin hệ thống.

V.4 Cấu trúc thư mục dạng đồ thị không chứa chu trình

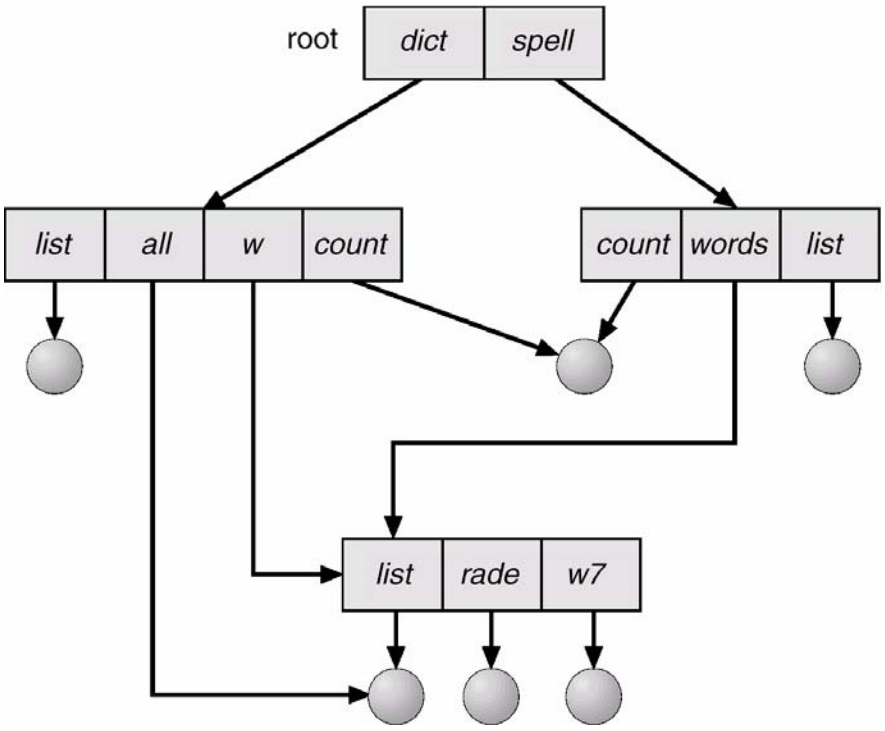
Xét hai người lập trình đang làm việc trên một dự án chung. Các tập tin gắn với dự án đó có thể được lưu trong thư mục con, tách rời chúng từ các dự án khác và các tập tin của hai người lập trình. Nhưng vì cả hai người lập trình có trách nhiệm ngang nhau trong dự án, cả hai muốn thư mục con ở trong các thư mục của chính họ. Thư mục con nên được chia sẻ. Một thư mục hay tập tin sẽ tồn tại trong hệ thống tập tin trong hai (hay nhiều hơn) nơi tại một thời điểm.

Cấu trúc cây ngăn cản việc chia sẻ các tập tin và thư mục. Một **đồ thị không chứa chu trình** (acyclic graph) cho phép thư mục chia sẻ thư mục con và tập tin (hình

IX-8). Cùng tập tin và thư mục con có thể ở trong hai thư mục khác nhau. Một đồ thị không chứa chu trình là trường hợp tổng quát của cơ chế thư mục có cấu trúc cây.

Một tập tin (hay thư mục) được chia sẻ không giống như hai bản sao của một tập tin. Với hai bản sao, mỗi người lập trình có thể thích hiển thị bản sao hơn bản gốc, nhưng nếu một người lập trình thay đổi nội dung tập tin, những thay đổi sẽ không xuất hiện trong bản sao của người còn lại. Với một tập tin được chia sẻ, chỉ một tập tin thực sự tồn tại vì thế bất cứ sự thay đổi được thực hiện bởi một người này lập tức nhìn thấy bởi người dùng khác. Việc chia sẻ là rất quan trọng cho các thư mục con; một tập tin mới được tạo bởi người này sẽ tự động xuất hiện trong tất cả thư mục con được chia sẻ.

Khi nhiều người đang làm việc như một nhóm, tất cả tập tin họ muốn chia sẻ có thể đặt vào một thư mục. Các UFD của tất cả thành viên trong nhóm chứa thư mục của tập tin được chia sẻ như một thư mục con. Ngay cả khi có một người dùng, tổ chức tập tin của người dùng này yêu cầu rằng một số tập tin được đặt vào các thư mục con khác nhau. Thí dụ, một chương trình được viết cho một dự án nên đặt trong thư mục của tất cả chương trình và trong thư mục cho dự án đó.



Hình 0-8 cấu trúc đồ thị không chứa chu trình

Các tập tin và thư mục con được chia sẻ có thể được cài đặt trong nhiều cách. Cách thông dụng nhất được UNIX dùng là tạo một mục từ thư mục được gọi là liên kết. Một liên kết là một con trỏ chỉ tới một tập tin hay thư mục con khác. Thí dụ, một liên kết có thể được cài đặt như tên đường dẫn tuyệt đối hay tương đối. Khi một tham chiếu tới tập tin được thực hiện, chúng ta tìm kiếm thư mục. Nếu mục từ thư mục được đánh dấu như một liên kết thì tên tập tin thật sự (hay thư mục) được cho. Chúng ta phân giải liên kết bằng cách sử dụng tên đường dẫn để định vị tập tin thật sự. Những liên kết được xác định dễ dàng bởi định dạng trong mục từ thư mục và được định rõ bằng các con trỏ gián tiếp. Hệ điều hành bỏ qua các liên kết này khi duyệt qua cây thư mục để lưu giữ cấu trúc không chứa chu trình của hệ thống.

Một tiếp cận khác để cài đặt các tập tin được chia sẻ là nhân bản tất cả thông tin về chúng trong cả hai thư mục chia sẻ. Do đó, cả hai mục từ là giống hệt nhau. Một liên kết rất khác từ mục từ thư mục gốc. Tuy nhiên, nhân bản mục từ thư mục làm cho bản gốc và bản sao không khác nhau. Một vấn đề chính với nhân bản mục từ thư mục là duy trì tính không đổi nếu tập tin bị sửa đổi.

Một cấu trúc thư mục đồ thị không chứa chu trình linh hoạt hơn cấu trúc cây đơn giản nhưng nó cũng phức tạp hơn. Một số vấn đề phải được xem xét cẩn thận. Một tập tin có nhiều tên đường dẫn tuyệt đối. Do đó, các tên tập tin khác nhau có thể tham chiếu tới cùng một tập tin. Trường hợp này là tương tự như vấn đề bí danh cho các ngôn ngữ lập trình. Nếu chúng ta đang cố gắng duyệt toàn bộ hệ thống tập tin-để tìm một tập tin, để tập hợp các thông tin thống kê trên tất cả tập tin, hay chép tất cả tập tin tới thiết bị lưu dự phòng-vấn đề này trở nên lớn vì chúng ta không muốn duyệt các cấu được chia sẻ nhiều hơn một lần.

Một vấn đề khác liên quan đến việc xoá. Không gian được cấp phát tới tập tin được chia sẻ bị thu hồi và sử dụng lại khi nào? một khả năng là xoá bỏ tập tin bất cứ khi nào người dùng xoá nó, nhưng hoạt động này để lại con trỏ chỉ tới một tập tin không tồn tại. Trong trường hợp xấu hơn, nếu các con trỏ tập tin còn lại chứa địa chỉ đĩa thật sự và không gian được dùng lại sau đó cho các tập tin khác, các con trỏ này có thể chỉ vào phần giữa của tập tin khác.

Trong một hệ thống mà việc chia sẻ được cài đặt bởi **liên kết biểu tượng**, trường hợp này dễ dàng quản lý hơn. Việc xoá một liên kết không cần tác động tập tin nguồn, chỉ liên kết bị xoá. Nếu chính tập tin bị xoá, không gian cho tập tin này được thu hồi, để lại các liên kết chơi vơi. Chúng ta có thể tìm các liên kết này và xoá chúng, nhưng nếu không có danh sách các liên kết được nối kết, việc tìm kiếm này sẽ tốn rất nhiều chi phí. Một cách khác, chúng ta có thể để lại các liên kết này cho đến khi nó được truy xuất. Tại thời điểm đó, chúng ta xác định rằng tập tin của tên được cho bởi liên kết không tồn tại và có thể bị lỗi để phục hồi tên liên kết; truy xuất này được đối xử như bất cứ tên tập tin không hợp lệ khác. Trong trường hợp UNIX, các liên kết biểu tượng được để lại khi một tập tin bị xoá và nó cho người dùng nhận thấy rằng tập tin nguồn đã mất hay bị thay thế. Microsoft Windows (tất cả ấn bản) dùng cùng tiếp cận.

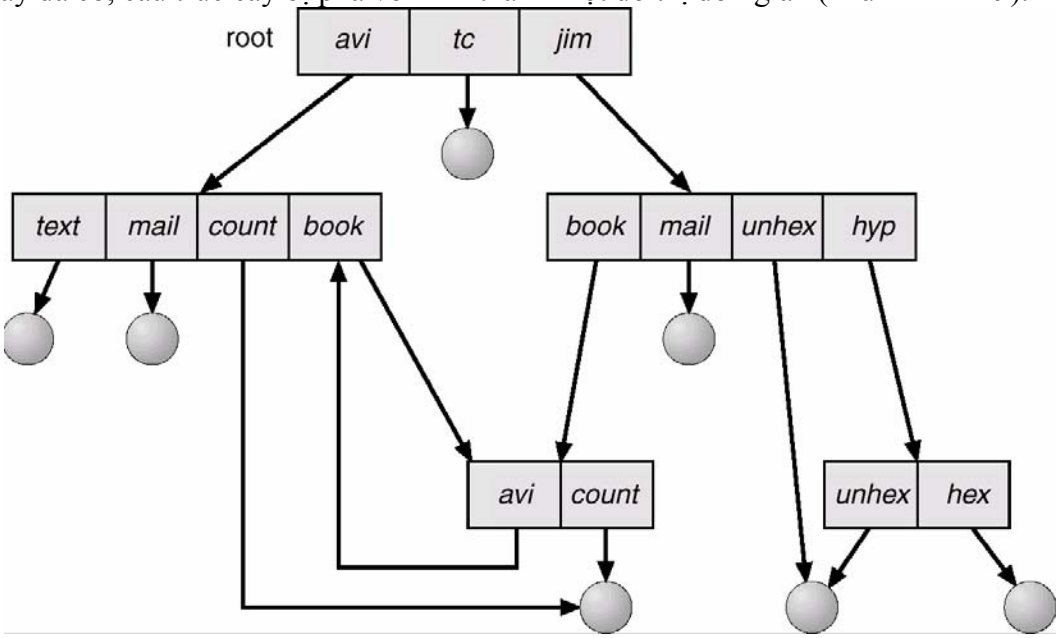
Một tiếp cận khác đối với việc xoá là giữ lại tập tin cho tới khi tất cả tham chiếu tới nó bị xoá. Để cài đặt tiếp cận này, chúng ta phải có một số cơ chế để xác định rằng tham chiếu cuối cùng tới tập tin bị xoá. Chúng ta giữ danh sách của tất cả tham chiếu tới một tập tin (các mục từ thư mục hay các liên kết biểu tượng). Khi một liên kết hay bản sao của mục từ thư mục được thiết lập, một mục từ mới được thêm tới danh sách tham chiếu tập tin. Khi một mục từ thư mục hay liên kết bị xoá, chúng ta gỡ bỏ mục từ của nó trên danh sách. Tập tin này bị xoá khi danh sách tham chiếu tập tin của nó là rỗng.

Trở ngại với tiếp cận này là kích thước của danh sách tham chiếu thay đổi và có thể rất lớn. Tuy nhiên, chúng ta thật sự không cần giữ toàn bộ danh sách-chúng ta chỉ cần giữ số đếm của số tham chiếu. Một liên kết mới hay mục từ thư mục mới sẽ tăng số đếm tham chiếu; xoá một liên kết hay mục từ sẽ giảm số đếm. Khi số đếm là 0, tập tin có thể được xoá; không còn tham chiếu nào tới nó. Hệ điều hành UNIX dùng tiếp cận này cho các liên kết không biểu tượng (hay liên kết cứng), giữ một số đếm tham chiếu trong khối thông tin tập tin (hay inode). Bằng cách ngăn cản hiệu quả nhiều tham chiếu tới các thư mục, chúng ta duy trì cấu trúc đồ thị không chứa chu trình.

Để tránh vấn đề này, một số hệ thống không cho phép thư mục hay liên kết được chia sẻ. Thí dụ, trong MS-DOS, cấu trúc thư mục là một cấu trúc cây hơn là đồ thị không chứa chu trình.

V.5 Cấu trúc thư mục dạng đồ thị tổng quát

Một vấn đề lớn trong việc dùng cấu trúc đồ thị không chứa chu trình là đảm bảo rằng không có chu trình trong đồ thị. Nếu chúng ta bắt đầu với thư mục hai cấp và cho phép người dùng tạo thư mục con, một thư mục có cấu trúc cây tạo ra. Dễ thấy rằng thêm các tập tin và thư mục con mới tới một thư mục có cấu trúc cây đã có vẫn bảo đảm tính tự nhiên của cấu trúc cây. Tuy nhiên, khi chúng ta liên kết một thư mục cấu trúc cây đã có, cấu trúc cây bị phá vỡ hình thành một đồ thị đơn giản (như hình IX-9).



Hình 0-9 thư mục đồ thị tổng quát

Một lợi điểm chính của đồ thị không chứa chu trình là tương đối đơn giản trong giải thuật duyệt đồ thị và xác định khi không có tham chiếu nữa tới tập tin. Chúng ta muốn tránh duyệt các phần được chia sẻ của đồ thị không chứa chu trình hai lần để tăng năng lực. Nếu chúng ta chỉ tìm một thư mục con được chia sẻ cho một tập tin xác định, chúng ta muốn tránh việc tìm kiếm thư mục con đó một lần nữa; tìm kiếm lần hai sẽ lãng phí thời gian.

Nếu các chu trình được cho phép tồn tại trong thư mục, chúng ta muốn tránh tìm kiếm bất cứ thành phần nào hai lần vì tính đúng đắn cũng như năng lực. Một giải thuật được thiết kế nghèo nàn dẫn tới vòng lặp vô tận trên các chu trình. Một giải pháp là giới hạn số lượng thư mục sẽ được truy xuất trong quá trình tìm kiếm.

Một vấn đề tương tự tồn tại khi chúng ta cố gắng xác định khi nào một tập tin có thể bị xoá. Như với cấu trúc thư mục đồ thị không chứa chu trình, một giá trị 0 trong số đếm tham chiếu có nghĩa là không còn tham chiếu nữa tới tập tin hay thư mục và tập tin có thể bị xoá. Tuy nhiên, khi có chu trình tồn tại, số đếm tham chiếu khác 0 ngay cả khi không còn tham chiếu tới thư mục hay tập tin. Sai sót này dẫn tới khả năng tham chiếu chính nó (hay chu trình) trong cấu trúc thư mục. Trong trường hợp này, chúng ta cần dùng cơ chế thu dọn rác (garbage colection) để xác định khi tham chiếu cuối cùng bị xoá và không gian đĩa có thể được cấp phát lại. Thu dọn rác

liên quan đến việc duyệt toàn bộ hệ thống tập tin, đánh dấu mọi thứ có thể được truy xuất. Sau đó, duyệt lần hai tập hợp mọi thứ không được đánh dấu trên danh sách không gian trống. Tuy nhiên, thu dọn rác cho một đĩa dựa trên hệ thống tập tin rất mất thời gian và do đó hiếm khi được thực hiện.

Thu dọn rác cần thiết chỉ vì có chu trình trong đồ thị. Do đó, cấu trúc đồ thị không chứa chu trình dễ hơn nhiều. Khó khăn là tránh chu trình khi các liên kết mới được thêm vào cấu trúc. Chúng ta biết như thế nào và khi nào một liên kết mới sẽ hình thành chu trình? Có nhiều giải thuật phát hiện các chu trình trong đồ thị; tuy nhiên chi phí tính toán cao, đặc biệt khi đồ thị ở trên đĩa lưu trữ. Một giải thuật đơn giản hơn trong trường hợp đặc biệt của thư mục và liên kết là bỏ qua liên kết khi duyệt qua thư mục. Chu trình có thể tránh được và không có chi phí thêm xảy ra.

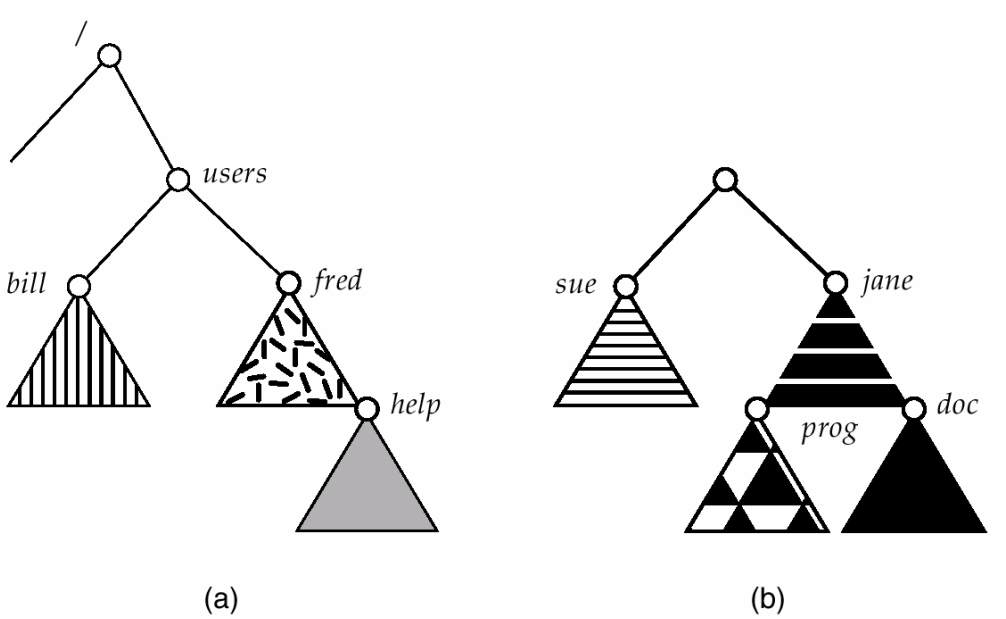
VI Gắn hệ thống tập tin

Giống như một tập tin phải được mở trước khi nó được dùng, một hệ thống tập tin phải được ***gắn vào*** (mounted) trước khi nó có thể sẵn dùng cho các quá trình trên hệ thống. Đặc biệt hơn, cấu trúc thư mục có thể được xây dựng trên nhiều phân khu, mà phải được gắn vào để làm cho chúng sẵn dùng trong không gian tên hệ thống tập tin.

Thủ tục gắn vào không phức tạp. Hệ điều hành được cho tên của thiết bị và vị trí trong cấu trúc tập tin tại nơi nó được gán vào hệ thống tập tin (điểm gắn-mount point). Điển hình, một điểm gắn thường là thư mục rỗng nơi hệ thống tập tin sẽ được gắn vào. Thí dụ, trên hệ thống UNIX, một hệ thống tập tin chứa các thư mục dành riêng cho người dùng (home directory) có thể được gắn vào như /home; sau đó để truy xuất tới cấu trúc tập tin trong hệ thống tập tin đó, người dùng có thể đến trước các tên thư mục /home, như trong /home/jane. Gắn vào hệ thống tập tin đó dưới /users sẽ dẫn tới tên đường dẫn /users/jane để đạt cùng thư mục.

Tiếp theo, hệ điều hành kiểm tra thiết bị chứa một hệ thống tập tin hợp lệ không. Nó thực hiện như thế bằng cách yêu cầu trình điều khiển thiết bị đọc thư mục thiết bị và kiểm tra thư mục có định dạng như mong muốn. Cuối cùng, hệ điều hành ghi nhận trong cấu trúc thư mục của nó rằng hệ thống tập tin được gắn vào tại điểm gắn xác định. Cơ chế này cho phép hệ điều hành duyệt cấu trúc thư mục của nó, chuyển qua lại giữa các hệ thống tập tin một cách hợp lý.

Để hiển thị việc gắn tập tin, xem xét hệ thống tập tin được mô tả như hình IX-10, ở đây hình này hình tam giác hiển thị cây con của các thư mục đang quan tâm. Trong hình (a) hiển thị hệ thống tập tin đã có, trong hình (b) hiển thị một phân khu chưa được gắn vào /device/dsk. Tại điểm này, chỉ các tập tin trên hệ thống tập tin đã tồn tại mới có thể được truy xuất. Trong hình IX-11, hiển thị các ảnh hưởng của việc gắn vào phân khu này trên /device/dsk qua /users. Nếu phân khu này chưa được gắn, hệ thống tập tin được phục hồi tới trường hợp được mô tả như hình IX-10.

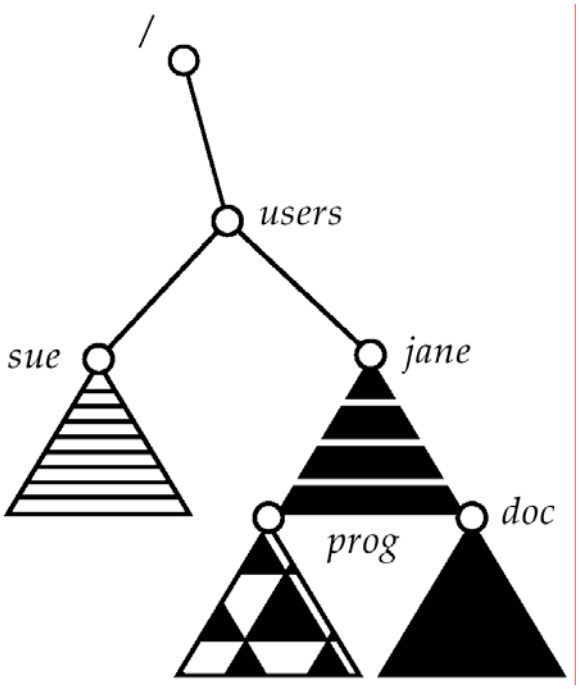


Hình 0-10 Hệ thống tập tin. (a) đã tồn tại. (b) phân khu chưa được gắn

Hệ thống này áp đặt các ngữ nghĩa để phân cấp chức năng. Thí dụ, một hệ thống không được phép vượt qua một thư mục chứa các tập tin hay làm hệ thống tập tin được gắn vào sẵn dùng tại thư mục đó và che đậy các tập tin đã có của thư mục cho đến khi hệ thống tập tin được gỡ ra (unmounted), kết thúc việc sử dụng hệ thống tập tin và cho phép truy xuất tới các tập tin trong thư mục đó.

Xét các hoạt động của hệ điều hành Macintosh. Bất cứ khi nào hệ thống gặp một đĩa tại thời điểm đầu tiên (đĩa cứng tại thời điểm khởi động hay đĩa mềm tại thời điểm chèn đĩa vào ổ), hệ điều hành Macintosh tìm kiếm một hệ thống tập tin cho thiết bị đó. Nếu nó tìm thấy một hệ thống tập tin, nó tự động gắn hệ thống tập tin tại mức gốc, thêm biểu tượng thư mục trên màn hình được ghi nhãn với tên của hệ thống tập tin (như được lưu trong thư mục thiết bị). Sau đó, người dùng có thể nhấp vào biểu tượng để hiển thị hệ thống tập tin vừa được gắn vào.

Họ hệ điều hành Microsoft Windows (95, 98, NT, 2000) duy trì một cấu trúc thư mục hai cấp mở rộng với các thiết bị và phân khu được gán một ký tự ổ đĩa. Các phân khu có cấu trúc thư mục đồ thị tổng quát được gắn liền với ký tự ổ đĩa. Sau đó, đường dẫn tới một tập tin xác định có dạng ***ký tự ổ đĩa:\đường dẫn\tới\tập tin***. Những hệ điều hành này tự phát hiện tất cả thiết bị và gắn vào tất cả hệ thống tập tin được định vị tại thời điểm khởi động. Trong một vài hệ thống, như UNIX, các lệnh gắn vào là hiện (explicit). Một tập tin cấu hình hệ thống chứa danh sách thiết bị và các điểm gắn vào để tự động gắn vào tại thời điểm khởi động, nhưng những gắn vào khác phải được thực hiện thủ công.



Hình 0-11 Điểm gắn vào

VII Chia sẻ tập tin

VII.1 Nhiều người dùng

Khi hệ điều hành cung cấp nhiều người dùng, các vấn đề chia sẻ tập tin, đặt tên tập tin, và bảo vệ tập tin trở nên quan trọng. Đối với một cấu trúc thư mục cho phép các tập tin được chia sẻ bởi nhiều người dùng, hệ thống phải dàn xếp việc chia sẻ tập tin. Mặc định, hệ thống có thể cho phép một người dùng truy xuất các tập tin của người dùng khác hay nó yêu cầu rằng một người dùng gán quyền truy xuất cụ thể tới các tập tin.

Để cài đặt chia sẻ và bảo vệ, hệ thống phải duy trì nhiều thuộc tính tập tin và thư mục hơn trên hệ thống đơn người dùng. Mặc dù, có nhiều tiếp cận cho chủ đề này, hầu hết các hệ thống đưa ra khái niệm người sở hữu (owner) và nhóm (group) tập tin/thư mục. Người sở hữu là người dùng có thể thay đổi các thuộc tính, gán truy xuất, và có hầu hết điều khiển qua tập tin và thư mục. Thuộc tính nhóm của tập tin được dùng để định nghĩa tập hợp con các người dùng có thể chia sẻ truy xuất tới tập tin.

VII.2 Hệ thống tập tin ở xa

Sự phát triển của mạng cho phép giao tiếp giữa các máy tính ở xa. Mạng cho phép chia sẻ các tài nguyên trải rộng trong một khu hay thậm chí khắp thế giới. Một tài nguyên quan trọng để chia sẻ là dữ liệu ở dạng tập tin. Thông quan sự phát triển mạng và công nghệ tập tin, phương pháp chia sẻ tập tin thay đổi. Trong phương pháp đầu tiên được cài đặt, người dùng truyền tập tin giữa các máy tính bằng chương trình gọi là ftp. Phương pháp quan trọng thứ hai là hệ thống tập tin phân tán (distributed file system-DFS) trong đó, các thư mục ở xa có thể được nhìn thấy từ máy cục bộ. Trong một số cách, phương pháp thứ ba, World Wide Web là một sự trở lại của phương

pháp đầu tiên. Một trình duyệt được yêu cầu để đạt được truy xuất các tập tin từ xa và các thao tác riêng biệt được dùng để truyền tập tin.

VIII Bảo vệ

Khi thông tin được giữ trong một hệ thống máy tính, chúng ta muốn giữ nó an toàn từ hỏng hóc vật lý (khả năng tin cậy) và những truy xuất không hợp lý (bảo vệ). Khả năng tin cậy thường được cung cấp bởi nhân bản các tập tin. Nhiều máy tính có các chương trình hệ thống tự động chép các tập tin trên đĩa tới băng từ tại những khoảng thời gian đều đặn để duy trì một bản sao. Hệ thống tập tin có thể bị hỏng bởi phần cứng, thay đổi đột ngột về điện, nhiệt độ tăng cao,..các tập tin có thể bị xoá do rủi ro. Những con bọ (bugs) trong phần mềm hệ thống tập tin có thể làm cho nội dung tập tin bị mất.

Bảo vệ có thể được cung cấp trong nhiều cách. Đối với một hệ thống người dùng đơn nhỏ, chúng ta có thể cung cấp sự bảo vệ bằng cách gỡ bỏ các đĩa mềm và khoá chúng trong ngăn kéo. Tuy nhiên, trong hệ thống đa người dùng, những cơ chế khác được yêu cầu.

VIII.1 Các kiểu truy xuất

Nhu cầu bảo vệ tập tin là một kết quả trực tiếp của khả năng để truy xuất tập tin. Hệ thống không cho phép truy xuất các tập tin của người dùng khác thì không cần bảo vệ. Do đó, chúng ta có thể cung cấp sự bảo vệ toàn diện bằng cách cấm truy xuất. Một cách khác, chúng ta có thể cung cấp truy xuất thoải mái và không cần bảo vệ. Cả hai tiếp cận là quá cực đoan cho các sử dụng thông thường. Yêu cầu truy xuất được kiểm soát là gì?

Các cơ chế bảo vệ cung cấp truy xuất được kiểm soát bằng cách giới hạn kiểu truy xuất tập tin có thể thực hiện. Truy xuất được phép hay bị từ chối phụ thuộc nhiều yếu tố, một trong những yếu tố là kiểu truy xuất được yêu cầu. Nhiều kiểu thao tác có thể được kiểm soát:

- Đọc (Read): đọc từ tập tin.
- Viết (Write): viết hay viết lại tập tin.
- Thực thi (Execute): nạp tập tin vào bộ nhớ và thực thi nó.
- Chèn cuối (Append): viết thông tin mới vào cuối tập tin.
- Xoá (Delete): xoá tập tin và giải phóng không gian để có thể dùng lại
- Liệt kê (List): liệt kê tên và thuộc tính của tập tin

Những thao tác khác như đổi tên, chép, soạn thảo tập tin có thể cũng được kiểm soát. Tuy nhiên, đối với nhiều hệ thống, các chức năng cao hơn này có thể được cài đặt bởi một chương trình hệ thống thực hiện lời gọi hệ thống cấp thấp hơn. Bảo vệ được cung cấp chỉ tại cấp thấp hơn. Thí dụ, chép một tập tin có thể được cài đơn giản bởi một chuỗi các yêu cầu đọc. Trong trường hợp này, người dùng với truy xuất đọc cũng có thể làm cho tập tin được chép, in,..

Nhiều cơ chế bảo vệ được đề nghị. Mỗi cơ chế có lợi điểm và nhược điểm và phải phù hợp cho ứng dụng được dự định. Một hệ thống máy tính nhỏ được dùng chỉ bởi một vài thành viên của một nhóm nghiên cứu có thể không cần cùng kiểu bảo vệ như máy tính của công ty lớn.

VIII.2 Kiểm soát truy xuất

Tiếp cận thông dụng nhất đối với vấn đề bảo vệ là thực hiện truy xuất phụ thuộc định danh của người dùng. Những người dùng khác nhau cần các kiểu truy xuất khác nhau tới một tập tin hay thư mục. Cơ chế thông dụng nhất để cài đặt truy xuất phụ thuộc định danh là gắn với mỗi tập tin và thư mục một danh sách kiểm soát truy xuất (access-control list-ACL) xác định tên người dùng và kiểu truy xuất được phép cho mỗi người dùng. Khi một người dùng yêu cầu truy xuất tới một tập tin cụ thể, hệ điều hành kiểm tra danh sách truy xuất được gắn tới tập tin đó. Nếu người dùng đó được liệt kê cho truy xuất được yêu cầu, truy xuất được phép. Ngược lại, sự vi phạm bảo vệ xảy ra và công việc của người dùng đó bị từ chối truy xuất tới tập tin.

Tiếp cận này có lợi điểm của việc cho phép các phương pháp truy xuất phức tạp. Vấn đề chính với các danh sách truy xuất là chiều dài của nó. Nếu chúng ta muốn cho phép mọi người dùng đọc một tập tin, chúng ta phải liệt kê tất cả người dùng với truy xuất đọc. Kỹ thuật này có hai kết quả không mong muốn:

- Xây dựng một danh sách như thế có thể là một tác vụ dài dòng và không đáng, đặc biệt nếu chúng ta không biết trước danh sách người dùng trong hệ thống.
- Mục từ thư mục, trước đó có kích thước cố định, bây giờ có kích thước thay đổi, dẫn đến việc quản lý không gian phức tạp hơn

Những vấn đề này có thể được giải quyết bởi việc dùng ấn bản cô đọng của danh sách truy xuất.

Để cô đọng chiều dài của danh sách kiểm soát truy xuất, nhiều hệ thống nhận thấy 3 sự phân cấp người dùng trong nối kết với mỗi tập tin:

- Người sở hữu (Owner): người dùng tạo ra tập tin đó
- Nhóm (Group): tập hợp người dùng đang chia sẻ tập tin và cần truy xuất tương tự là nhóm hay nhóm làm việc
- Người dùng khác (universe): tất cả người dùng còn lại trong hệ thống

Tiếp cận phổ biến gần đây nhất là kết hợp các danh sách kiểm soát truy xuất với người sở hữu, nhóm và cơ chế kiểm soát truy xuất được mô tả ở trên

Để cơ chế này làm việc hợp lý, các quyền và danh sách truy xuất phải được kiểm soát chặt chẽ. Kiểm soát này có thể đạt được trong nhiều cách. Thí dụ, trong hệ thống UNIX, các nhóm có thể được tạo và sửa đổi chỉ bởi người quản lý của tiện ích. Do đó, kiểm soát này đạt được thông qua giao tiếp người dùng.

Với việc phân cấp bảo vệ được giới hạn hơn, chỉ có ba trường được yêu cầu để xác định bảo vệ. Mỗi trường thường là một tập hợp các bit, mỗi trường cho phép hay ngăn chặn truy xuất được gắn với nó. Thí dụ, hệ thống UNIX định nghĩa 3 trường 3 bit-rwx, ở đây r kiểm soát truy xuất đọc, w kiểm soát truy xuất viết, x kiểm soát truy xuất thực thi. Một trường riêng rẻ được giữ cho người sở hữu, cho nhóm tập tin và cho tất cả người dùng khác. Trong cơ chế này, 9 bits trên tập tin được yêu cầu để ghi lại thông tin bảo vệ.

VIII.3 Các tiếp cận bảo vệ khác

Một tiếp cận khác cho vấn đề bảo vệ là gắn mật khẩu với mỗi tập tin. Giống như truy xuất tới hệ thống máy tính thường được kiểm soát bởi một mật khẩu, truy xuất tới mỗi tập tin có thể được kiểm soát bởi một mật khẩu. Nếu các mật khẩu được chọn một cách ngẫu nhiên và thường được thay đổi thì cơ chế này có thể hiệu quả trong truy xuất có giới hạn tới tập tin cho những người dùng biết mật khẩu. Tuy nhiên, cơ chế

này có nhiều nhược điểm. Thứ nhất, số lượng mật khẩu mà người dùng cần nhớ quá nhiều, làm cho cơ chế này không thực tế. Thứ hai, nếu chỉ một mật khẩu được dùng cho tất cả tập tin thì một khi nó bị phát hiện tất cả tập tin có thể truy xuất. Một số hệ thống cho phép người dùng gắn một mật khẩu tới một thư mục con hơn là với từng tập tin riêng rẻ để giải quyết vấn đề này. Thứ ba, thường chỉ một mật khẩu gắn với tất cả tập tin người dùng. Do đó, bảo vệ dựa trên cơ sở tất cả hay không có gì (all-or-nothing). Để cung cấp sự bảo vệ trên cấp độ chi tiết hơn chúng ta phải dùng nhiều mật khẩu.

IX Tóm tắt

Một tập tin là một kiểu dữ liệu trừu tượng được định nghĩa và được cài đặt bởi hệ điều hành. Nó là một chuỗi các mẫu tin luận lý. Một mẫu tin luận lý có thể là một byte, một dòng (có chiều dài cố định hay thay đổi), hay có thành phần dữ liệu phức tạp hơn. Hệ điều hành có thể hỗ trợ nhiều kiểu mẫu tin khác nhau hay để sự hỗ trợ đó tới một chương trình ứng dụng.

Mỗi thiết bị trong một tập tin giữ một bảng volume nội dung hay thư mục thiết bị liệt kê vị trí các tập tin trên thiết bị. Ngoài ra, nó có ích để tạo các thư mục cho phép các tập tin được tổ chức trong thư mục đó. Một thư mục đơn cấp trong hệ thống đơn người dùng gây ra các vấn đề đặt tên vì mỗi tập tin phải có tên duy nhất. Thư mục hai cấp giải quyết vấn đề này bằng cách tạo một thư mục riêng cho mỗi người dùng. Mỗi người dùng có thư mục riêng, chứa tập tin riêng. Thư mục liệt kê các tập tin bằng tên và chứa những thông tin như vị trí tập tin trên đĩa, chiều dài, kiểu, người sở hữu, thời gian tạo, thời điểm dùng gần nhất,…

Tổng quát hóa tính tự nhiên của thư mục hai cấp là thư mục có cấu trúc cây. Thư mục có cấu trúc cây cho phép một người dùng tạo thư mục con để tổ chức các tập tin. Cấu trúc thư mục đồ thị không chứa chu trình cho phép các thư mục con và tập tin được chia sẻ nhưng tìm kiếm và xóa phức tạp. Một cấu trúc đồ thị tổng quát linh động hơn trong việc chia sẻ tập tin và thư mục, nhưng yêu cầu thu dọn rác để phục hồi không gian đĩa không được dùng.

Đĩa được phân chia thành một hay nhiều phân khu, mỗi phân khu chứa một hệ thống tập tin. Hệ thống tập tin này được gán vào cấu trúc đặt tên của hệ thống để làm cho chúng sẵn dùng. Cơ chế đặt tên khác nhau bởi các hệ điều hành khác nhau. Một khi được gán vào, các tập tin trong phân khu là sẵn dùng. Các hệ thống tập tin có thể được gỡ ra (unmount) để vô hiệu hóa truy xuất hay để bảo trì.

Chia sẻ tập tin phụ thuộc vào ngữ nghĩa được cung cấp bởi hệ thống. Các tập tin có nhiều người đọc, viết hay bị giới hạn việc chia sẻ. Hệ thống tập tin phân tán cho phép máy khách hàng gắn các phân khu hay thư mục vào từ nhiều server. Với điều kiện chúng có thể truy xuất nhau qua mạng. Các hệ thống tập tin ở xa có những thách thức về khả năng tin cậy, năng lực và bảo mật. Hệ thống thông tin được phân tán duy trì người dùng, máy chủ, thông tin truy xuất như khách hàng và thông tin trạng thái chia sẻ servers để quản lý việc sử dụng và truy xuất.

Vì các tập tin là cơ chế lưu trữ thông tin quan trọng trong hầu hết các hệ thống máy tính nên bảo vệ tập tin là cần thiết. Truy xuất tới các tập tin được kiểm soát riêng cho mỗi loại truy xuất-đọc, viết, thực thi, chèn cuối, xóa, liệt kê thư mục,..Bảo vệ tập tin có thể được cung cấp bởi mật khẩu, bởi danh sách truy xuất hay bởi những kỹ thuật phức tạp.